

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
«Установка пакетов в Python. Виртуальные окружения»**

**Отчет по лабораторной работе № 2.16
по дисциплине «Основы программной инженерии»**

Выполнил студент группы

ПИЖ-б-о-21-1

Трушева В. О. .« » 2023г.

Подпись студента _____

Работа защищена «_____ 20__ г.

Проверила Воронкин Р.А. _____

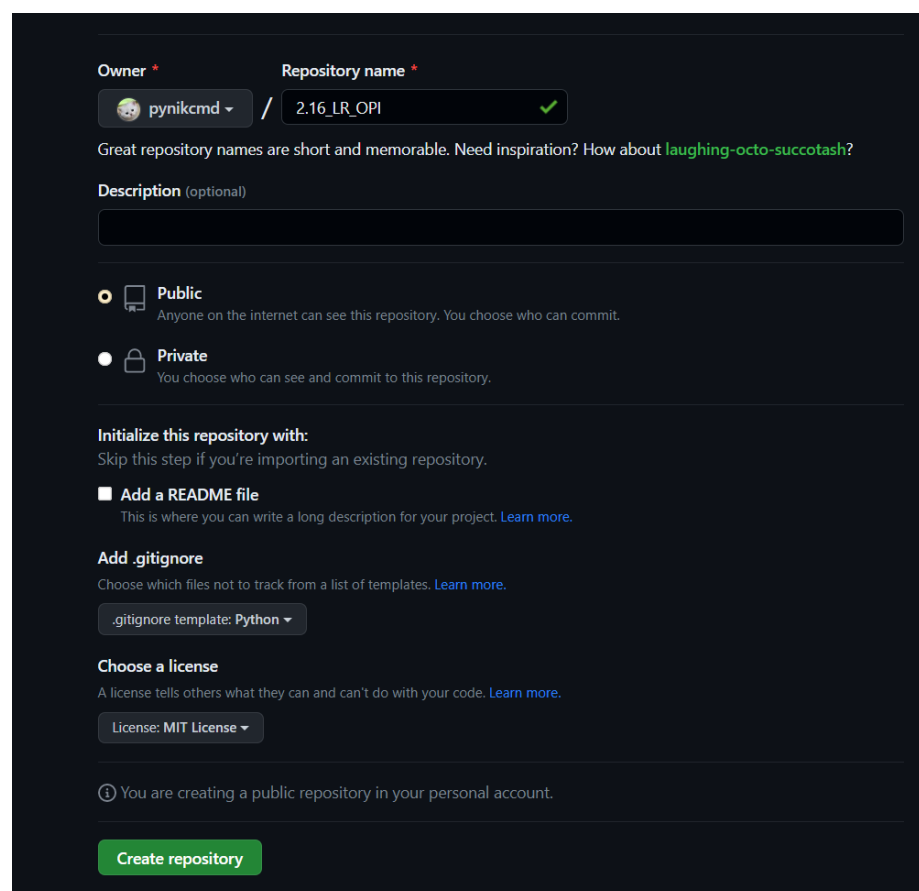
(подпись)

Ставрополь 2023

Цель работы: приобретение навыков по работе с данными формата JSON с помощью языка программирования Python версии 3.x.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



Owner * / Repository name *

Great repository names are short and memorable. Need inspiration? How about [laughing-octo-succotash?](#)

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License

ⓘ You are creating a public repository in your personal account.

Create repository

Рисунок – Создание репозитория

3. Выполните клонирование созданного репозитория.

```
D:\fgit>git clone https://github.com/pynikcmd/2.16_LR_OPI.git
Cloning into '2.16_LR_OPI'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (7/7), done.
Resolving deltas: 100% (1/1), done.
```

Рисунок – Клонирование репозитория

4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.

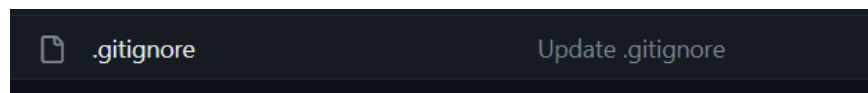


Рисунок – Дополнен gitignore

5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

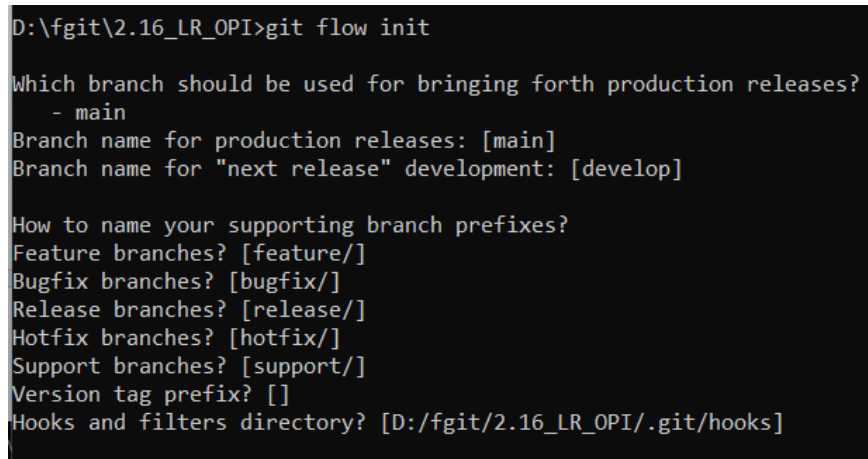


Рисунок – Модель git-flow

6. Создайте проект PyCharm в папке репозитория.

7. Проработайте примеры лабораторной работы. Создайте для них отдельные модули языка Python. Зафиксируйте изменения в репозитории.

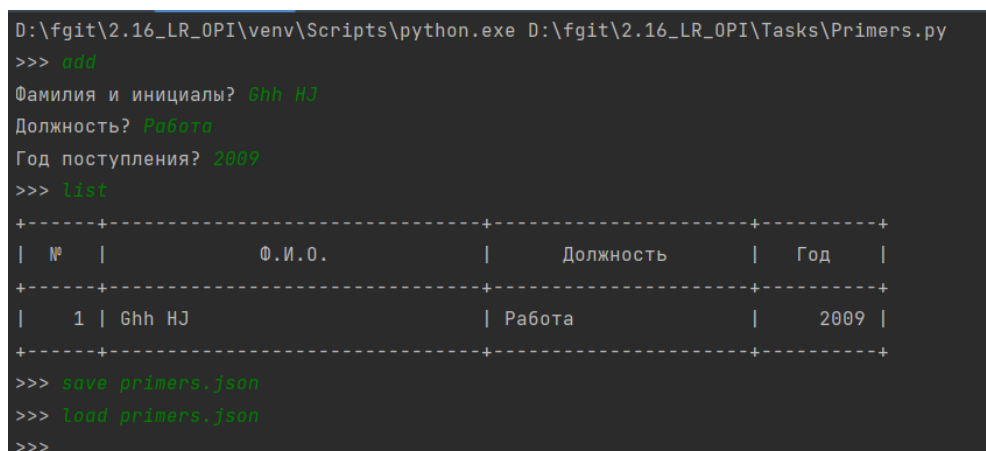


Рисунок – Результат работы программы

8. Приведите в отчете скриншоты результатов выполнения примера при различных исходных данных вводимых с клавиатуры.

```
select
>>> Неизвестная команда select
select 4
+-----+-----+-----+-----+
| № |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
|  1 | Ghh HJ                   |      Работа        |      2009     |
+-----+-----+-----+-----+
>>> help
Список команд:

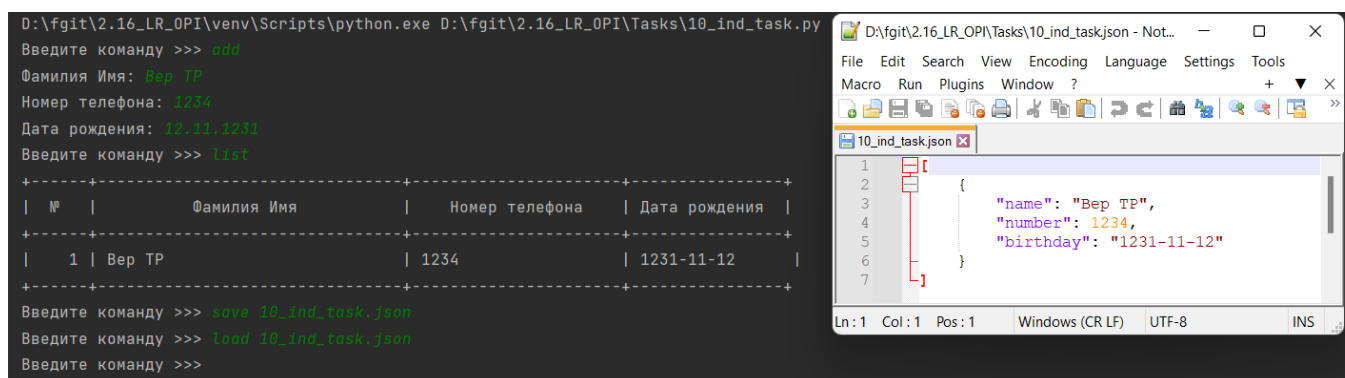
add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
load - загрузить данные из файла;
save - сохранить данные в файл;
exit - завершить работу с программой.
>>>
```

Рисунок – Результат работы программы

9. Зафиксируйте сделанные изменения в репозитории.

10. Приведите в отчете скриншоты работы программ решения индивидуальных заданий.

Условие. Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по датам рождения; вывод на экран информации о человеке, номер телефона которого введен с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.



```
D:\fgit\2.16_LR_OPI\venv\Scripts\python.exe D:\fgit\2.16_LR_OPI\Tasks\10_ind_task.py
Введите команду >>> add
Фамилия Имя: Bep TP
Номер телефона: 1234
Дата рождения: 12.11.1231
Введите команду >>> list
+-----+-----+-----+-----+
| № |          Фамилия Имя          |      Номер телефона      |      Дата рождения      |
+-----+-----+-----+-----+
|  1 | Bep TP                       |      1234                |      1231-11-12         |
+-----+-----+-----+-----+
Введите команду >>> save 10_ind_task.json
Введите команду >>> load 10_ind_task.json
Введите команду >>>
```

```
1
2
3
4 {
5   "name": "Bep TP",
6   "number": 1234,
7   "birthday": "1231-11-12"
8 }
```

Рисунок – Результат выполнения программы

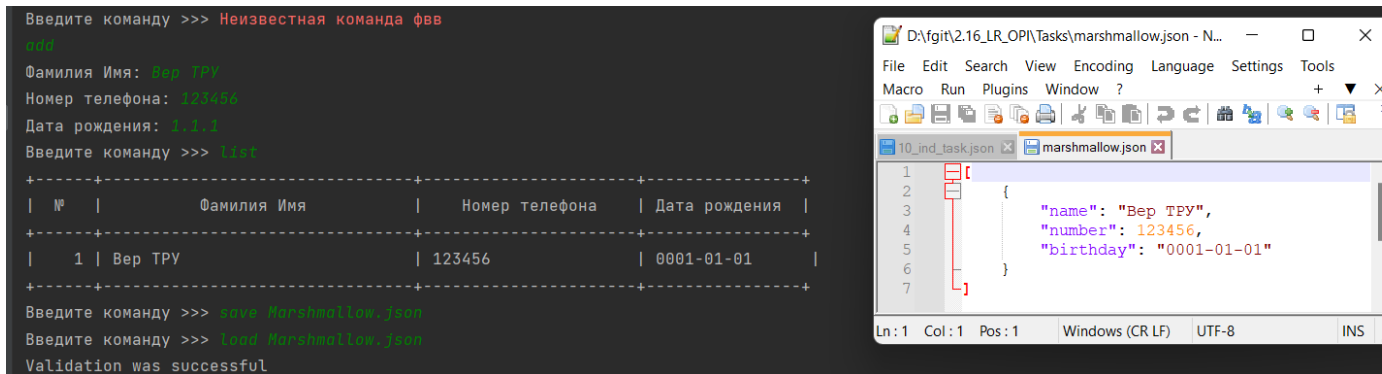


Рисунок – Результат выполнения программы (Marshmallow)

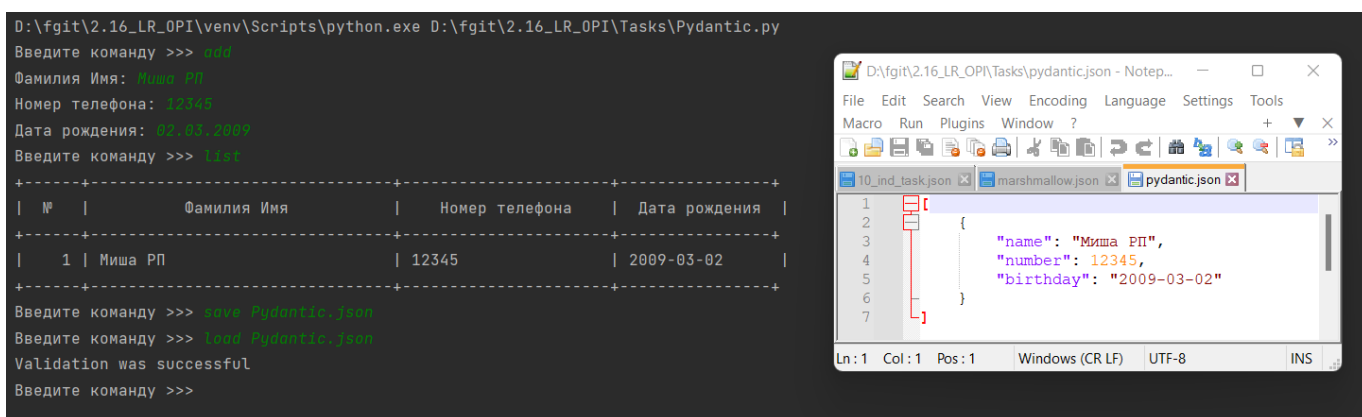


Рисунок – Результат выполнения программы (Pydantic)

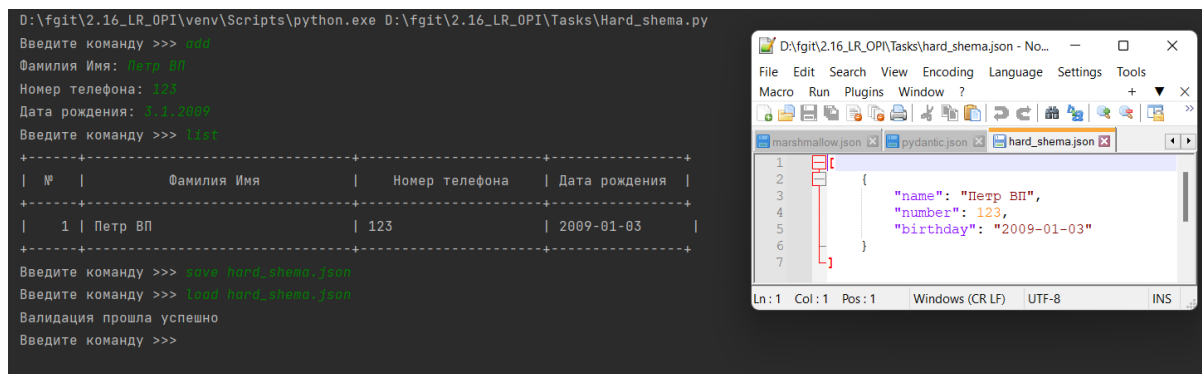


Рисунок – Результат выполнения программы (Hard)

11. Зафиксируйте сделанные изменения в репозитории.

12. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.

13. Выполните слияние ветки для разработки с веткой master/main.

14. Отправьте сделанные изменения на сервер GitHub.

15. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы:

1. Для чего используется JSON?

JSON - текстовый формат обмена данными, основанный на JavaScript. JSON используется в веб-разработке для передачи данных между клиентом и сервером.

2. Какие типы значений используются в JSON?

В JSON используются следующие типы значений:

Строки - последовательности символов, заключенные в двойные кавычки, например: "Hello, world!".

Числа - целые числа или числа с плавающей точкой, например: 42, 3.14.

Логические значения - true или false.

null - специальное значение, обозначающее отсутствие значения.

Массивы - упорядоченные списки значений, заключенные в квадратные скобки и разделенные запятыми, например: [1, 2, 3].

Объекты - неупорядоченные коллекции пар "ключ-значение", заключенные в фигурные скобки и разделенные запятыми, например: {"name": "John", "age": 30}.

3. Как организована работа со сложными данными в JSON?

В Python для работы со сложными данными в формате JSON используется стандартный модуль json. С помощью этого модуля можно преобразовывать данные из формата JSON в объекты Python и наоборот.

Вложенные объекты

Вложенные объекты в JSON - это объекты, которые содержат другие объекты в качестве своих свойств. Эти вложенные объекты могут иметь свои собственные свойства, которые также могут быть объектами.

В Python вложенные объекты JSON могут быть представлены в виде вложенных словарей. То есть каждый вложенный объект может быть представлен в виде словаря в Python, где ключи являются свойствами объекта, а значения являются соответствующими значениями свойств.

Вложенные массивы

Данные также могут быть вложены в формате JSON, используя JavaScript массивы, которые передаются как значения. JavaScript использует квадратные скобки [] для формирования массива. Массивы по своей сути — это упорядоченные коллекции и могут включать в себя значения совершенно разных типов данных. Мы можем использовать массив при работе с большим количеством данных, которые могут быть легко сгруппированы вместе, как например, если есть несколько разных сайтов и профайлов в социальных сетях ассоциированных с одним пользователем.

4. Самостоятельно ознакомьтесь с форматом данных JSON5? В чем отличие этого формата от формата данных JSON?

Формат обмена данными JSON5 — это расширенная JSON-версия, которая призвана смягчить некоторые ограничения JSON, расширив его синтаксис и включив в него некоторые функции из ECMAScript 5.1 (стандарт языка программирования JavaScript).

Некоторые нововведения:

- Поддерживаются как однострочные //, так и многострочные /* */ комментарии.
- Записи и списки могут иметь запятую после последнего элемента (удобно при копировании элементов).
- Строки могут заключаться как в одинарные, так и в двойные кавычки.

5. Какие средства языка программирования Python могут быть использованы для работы с данными в формате JSON5?

В Python для работы с данными в формате JSON5 можно использовать сторонние библиотеки, такие как json5 или demjson. Эти библиотеки предоставляют функции для чтения и записи данных в формате JSON5.

Чтения файла в формате JSON5 и преобразования его в объект Python:

```
with open('example.json5') as f:
```

```
    data = json5.load(f)
```

Также можно использовать `json5.dumps()` для преобразования объекта Python в формат JSON5 и записи его в файл:

```
data = {'foo': 'bar', 'baz': [1, 2, 3]}
```

```
with open('example.json5', 'w') as f:
```

```
    f.write(json5.dumps(data))
```

6. Какие средства предоставляет язык Python для сериализации данных в формате JSON?

Сериализация данных в формат JSON:

```
json.dump() # конвертировать python объект в json и записать в файл  
json.dumps() # тоже самое, но в строку.
```

7. В чем отличие функций `json.dump()` и `json.dumps()`?

`json.dumps()` конвертирует python объект в json и записывает его в строку вместо записи в файл.

8. Какие средства предоставляет язык Python для десериализации данных из формата JSON?

Десериализация данных из формата JSON:

`json.load()` # прочитать json из файла и конвертировать в python объект
`json.loads()` # тоже самое, но из строки с json (s на конце от string/строка).

9. Какие средства необходимо использовать для работы с данными формата JSON, содержащими кириллицу?

Использование кодировки UTF-8 или `ensure_ascii=False`

10. Самостоятельно ознакомьтесь со спецификацией JSON Schema? Что такое схема данных?

Схема данных - это формальное описание структуры и содержания данных, которые должны соответствовать определенным требованиям. Она определяет типы данных, формат, ограничения и правила валидации данных.

JSON Schema предоставляет механизм для описания схемы данных в формате JSON. С помощью JSON Schema можно задать ограничения на типы и форматы данных, допустимые значения, минимальные и максимальные значения и другие условия, которые должны быть выполнены для корректного представления данных.